

Introduction to Programming

with Java, for Beginners

Null Pointer Exception
Array use examples
2D Arrays

Recap

<code>int[] data;</code>	<i>data is a reference variable whose type is int[], meaning "array of ints". At this point its value is null.</i>
<code>data = new int[5];</code>	<i>The new operator causes a chunk of memory big enough for 5 ints to be allocated on the heap. Here, data is assigned a reference to the heap address.</i>
<code>data[0] = 6; data[1] = 10; data[2] = 12;</code>	<i>Initially, all five ints are 0. Here, three of them are assigned other values.</i>
<code>int[] info = {6, 10, 12, 0, 0};</code>	
<code>info = new int[]{6, 10, 12, 0, 0};</code>	

new type[] syntax

- The advantage of using the "new type[]" syntax is that it can be used in an assignment statement that is not a variable declaration statement
- E.g. `int [] data;`
`data = new int [] {5,10, 6};`
However:
`data = {5, 10, 6};` //will give an error

ESE112

2/15

Null Pointer Exception

- Suppose you *declare* a `int [] data` but you don't *define* it
 - Until you define `data`, it has the special value `null`
 - `null` is a legal value for any kind of object
 - > i.e. Person p, Counter c; Player mario
 - `null` can be assigned, tested, and printed
- But if you try to use a field or method of `null`, you get a *nullPointerException* i.e. you try to access some object that has not been created
 - `data[0]`
 - `p.getName()`
 - `mario.getStrength()`

ESE112

3/15

Error Checking with arrays

```
public static void printArray(int[ ]data){
    if (data == null || data.length == 0){
        System.out.println("Array is empty");
        return; //returns nothing
    }
    for (int i = 0; i < data.length; i++){
        System.out.println(data[i]);
    }
}
```

ESE112

4/15

Example of array use I

- Suppose you want to find the largest value in an array `scores` of 10 integers:

```
int largestScore = 0;
for (int i = 0; i < 10; i++) {
    if (scores[i] > largestScore) {
        largestScore = scores[i];
    }
}
```

- Would this code work if next time you had 12 scores? Or 8 scores?
- Do you see an error in the above program?

ESE112

5/15

Example of array use I (improved)

- To find the largest value in an array `scores` of (possibly negative) integers:

```
int largestScore = scores[0];
for (int i = 1; i < scores.length; i++) {
    if (scores[i] > largestScore) {
        largestScore = scores[i];
    }
}
```

ESE112

6/15

Example of array use II

- Suppose you want to find the location in which you find the largest value in an array `scores`

```
int largestScore = scores[0];
int index = 0;
for (int i = 1; i < scores.length; i++) {
    if (scores[i] > largestScore) {
        largestScore = scores[i];
        index = i;
    }
}
```

ESE112

7/15

2D Arrays

- Array can have 2, 3, or more dimensions
- When declaring a variable of such an array, use a pair of square brackets for each dimension
- For 2D arrays, the elements are indexed [row][column]
- Remember “RC” [row][column]

ESE112

8/15

Example 1: Table

- `int[][] table = new int[3][2];` or,
- `int[][] table = { {1, 2}, {3, 6}, {7, 8} };`
- For example, `table[1][1]` contains 6
- `table[2][1]` contains 8, and
- `table[1][2]` is “array out of bounds”

	0	1
0	1	2
1	3	6
2	7	8

ESE112

9/15

Processing 2D Arrays

- How to zero out this table on the previous slide ?

```
for (int i = 0; i < 3; i++){
    for (int j = 0; j < 2; j++){
        table[i][j] = 0;
    }
}
```

- Use a doubly-nested for-loop to process a 2D array
- In this example we know the number of rows (3) and columns (2)
- In general, it's better not to use “magic numbers” (here the 3 and 2) in the loop.
- How could this code be improved?

ESE112

10/15

Size of 2D Array

- `int[][] table = new int[3][2];`
- The length of this array is the number of rows: `table.length` is 3
- Each row contains an array
- To get the number of *columns*, pick a row and ask for its length: e.g. `table[0].length` is 2
 - Most of the time, you can assume all the rows are the same length

	0	1
0	1	2
1	3	6
2	7	8

ESE112

11/15

Example 2: Tic Tac Toe (TTT) Board

```
char[][] board;
board = new char[3][3]; // 3 rows, 3 cols
board[0][0] = 'O'; // row 0, column 0
board[1][1] = 'X'; // row 1, column 2
board[0][1] = 'X'; // row 0, column 1
...
```

ESE112

12/15

Example 2: Print TTT Board

```
char[][] data = new char[3][3];

//Printing 2D array example
for (int row = 0; row < 3; row++){
    for (int col = 0; col < 3; col++){
        System.out.print(data[row][col] + '\t');
    }
    System.out.println();
}
```

ESE112

13/15

Example 2: Generalized Printing

- This illustrates a general purpose way to print a 2D array
- It works even for “ragged” arrays, whose row lengths vary

```
public static void printArray(int[][] data){
    if (data == null || data.length == 0){
        System.out.println("Array is empty");
        return;
    }
    for (int row = 0; row < data.length; row++){
        for (int col = 0; col < data[row].length; col++){
            System.out.print(data[row][col] + "\t");
        }
        System.out.println();
    }
}
```

ESE112

14/15

Ragged Array

- Row lengths vary.
- Motivation: save space

```
> int[] one = {1,2,3}
> int[] two = {1,2,3,4,5,6}
> int[] three = {1,2};
>
> int[][] data = {one, two, three}
> data[0].length
3
> data[1].length
6
> data[2].length
2
> data[0] = three;
> data[1] = two;
> data[2] = one;
> data[0].length
2
> data[1].length
6
> data[2].length
3
```

ESE112

15/15